

```

library(reshape)
library(openair)
met <- read.csv(file="C:\\Users\\ahawki03\\Desktop\\beta_request\\data_from_AQS\\AMP501_1417934-0.txt",skip=0,header=T,sep="|")#scalar
met <- read.csv(file="C:\\Users\\ahawki03\\Desktop\\beta_request\\data_from_AQS\\AMP501_1418790-0.txt",skip=0,header=T,sep="|")#resultant
#a note on the resultant versus scalar current AERMET guidance says to use scalar, but old data is resultant. Not much of a difference between the two when the winds are not shifting around.
met <- met[met$Site.ID==9001,]#just grab the valley met data, needed for resultant winds
met1 <- data.frame(met[,6],met[,11],met[,12],met[,13])
colnames(met1) <- c("code","date","hour","parm")
met2 <- reshape(met1, idvar=c("date","hour"), timevar="code", direction="wide")
met2 <- met2[,1:4]
colnames(met2) <- c("date","hour","ws","wd")
conc <- read.csv(file="C:\\Users\\ahawki03\\Desktop\\beta_request\\data_from_AQS\\AMP501_1417948-0.txt",skip=0,header=T,sep="|")
conc1 <- (conc[conc$Site.ID==9001 | conc$Site.ID==9002,])
conc1 <- data.frame(conc1[,5],conc1[,11],conc1[,12],conc1[,13])
colnames(conc1) <- c("id","date","hour","so2")
conc2 <- reshape(conc1, idvar=c("date","hour"), timevar="id", direction="wide")
colnames(conc2) <- c("date","hour","valley","NW")
conc2 <- conc2[,1:4]
#merge(met2,conc2)
data <- merge(met2,conc2)
colnames(data) <- c("date","hour","ws","wd","valley","NW")
data1 <- data.frame(substr(data$date,1,4),substr(data$date,5,6),substr(data$date,7,8),data[,2:6])
colnames(data1) <- c("yr", "mo", "day", "hr", "ws", "wd","valley","NW")
data2 <- data.frame(as.POSIXct(paste(data1$yr,"-",data1$mo,"-",data1$day," ",data1$hr,:00",sep="")),format = "%Y-%m-%d %H:%M:%S"),data1[,5:8])
colnames(data2) <- c("date", "ws", "wd","valley","NW")

pdf(file="C:\\Users\\ahawki03\\Desktop\\beta_request\\data_from_AQS\\plots.pdf",width=11,height=8.5)
windRose(data1,angle=5,key.footer="knots",main="Windrose Valley 10m")

polarPlot(data1,pollutant='valley',units="knots",uncertainty = TRUE,resolution="fine")
polarPlot(data1,pollutant='NW',units="knots",uncertainty = TRUE,resolution="fine")

polarPlot(data1,pollutant='valley',units="knots",statistic="weighted.mean",resolution="fine",main="SO2 polarPlot (valley)")
polarPlot(data1,pollutant='NW',units="knots",statistic="weighted.mean",resolution="fine",main="SO2 polarPlot (NW)")

#pollutionRose(data1,pollutant='valley',angle=5,main="SO2 pollutionRose (Valley)")
#pollutionRose(data1,pollutant='NW',angle=5,main="SO2 pollutionRose (NW)")

polarFreq(data1,pollutant='valley',main="SO2 polarFreq (valley)",statistic="weighted.mean",min.bin=2,offset = 5, ws.int = 15, trans = FALSE)
polarFreq(data1,pollutant='NW',main="SO2 polarFreq (NW)",statistic="weighted.mean",min.bin=2,offset = 5, ws.int = 15, trans = FALSE)

pollutionRose(data1,pollutant='valley',angle=5,main="SO2 pollutionRose (valley)",breaks = c(0, 1, 3, 5, 10, 20, 40))
pollutionRose(data1,pollutant='NW',angle=5,main="SO2 pollutionRose (NW)",breaks = c(0, 1, 3, 5, 10, 20, 40))

```

```

polarCluster(data2, pollutant = "valley", n.clusters = 2:8,main="SO2 polarCluster (valley)")
polarCluster(data2, pollutant = "NW", n.clusters = 2:8,main="SO2 polarCluster (NW)")

#results <- polarCluster(data2, pollutant = "valley", n.clusters = 6)
#timeVariation(subset(results$data, cluster %in% c("1", "2")), pollutant = "valley",group = "cluster", col =
openColours("Paired", 6)[c(3, 4)])

#pdf(file="C:\\Users\\ahawki03\\Desktop\\beta_request\\data_from_AQS\\plots.pdf",width=11,height=8.5)
for (ii in 4:12){
  timePlot(selectByDate(data2, year=2015, month=ii),pollutant = c("valley","NW", "ws","wd"),main="Labadie SO2")
}
timePlot(selectByDate(data2, start = "27/8/2015", end = "28/8/2015"),pollutant = c("valley","NW", "ws","wd"))
timePlot(selectByDate(data2, start = "10/10/2015", end = "11/10/2015"),pollutant = c("valley","NW", "ws","wd"))
timePlot(selectByDate(data2, start = "8/11/2015", end = "10/11/2015"),pollutant = c("valley","NW", "ws","wd"))
dev.off()

camd<-
read.csv(file="C:\\Users\\ahawki03\\Desktop\\beta_request\\data_from_AQS\\Labadie2015.txt",skip=0,header=T,sep=",")
camd$date <- as.POSIXct(paste(camd$OP_DATE,camd$OP_HOUR),format="%m-%d-%Y %H")
camd1 <- reshape(camd, idvar=c("date"), timevar="UNITID", direction="wide")
camd2 <-
data.frame(camd1$date,camd1$SO2_MASS..lbs..1,camd1$SO2_MASS..lbs..2,camd1$SO2_MASS..lbs..3,camd1$SO2
_MASS..lbs..4,camd1$HEAT_INPUT..mmBtu..1,camd1$HEAT_INPUT..mmBtu..2,camd1$HEAT_INPUT..mmBtu..3,
camd1$HEAT_INPUT..mmBtu..4)
camd2$total <- rowSums(camd2[,2:5],na.rm=TRUE)
colnames(camd2) <- c("date","U1mass","U2mass","U3mass","U4mass","U1heat","U2heat","U3heat","U4heat","total")
data3 <- merge(data2,camd2)
write.table(data3,file="C:\\Users\\ahawki03\\Desktop\\beta_request\\data_from_AQS\\R7_summary.csv",sep=",")

#####

```

If you just want to view the raw data...

```

na.omit(data3[data3$NW>=0 & data3$valley>=0,]) #or change the number to something where both monitors > XXX
na.omit(data3[data3$NW>=30 | data3$valley>=30,]) # this is an "or"
#####

```

#lets take a look at the historic data

```

met <- read.csv(file="C:\\Users\\ahawki03\\Desktop\\beta_request\\data_from_AQS\\AMP501_1418786-
0.txt",skip=0,header=T,sep="|")#resultant
#a note on the resultant versus scalar current AERMET guidance says to use scalar, but old data is resultant. Not much
of a difference between the two when the winds are not shifting around.
met <- met[met$County.Code==183 & met$Site.ID==10,]#just grab the Augusta met data, needed for resultant winds
met1 <- data.frame(met[,6],met[,11],met[,12],met[,13])
colnames(met1) <- c("code","date","hour","parm")
met2 <- reshape(met1, idvar=c("date","hour"), timevar="code", direction="wide")
met2 <- met2[,1:4]
colnames(met2) <- c("date","hour","ws","wd")
conc <- read.csv(file="C:\\Users\\ahawki03\\Desktop\\beta_request\\data_from_AQS\\AMP501_1418821-
0.txt",skip=0,header=T,sep="|")
conc1 <- (conc[conc$County.Code=="183" & conc$Site.ID=="0010",])
conc1 <- data.frame(conc1[,5],conc1[,11],conc1[,12],conc1[,13])
colnames(conc1) <- c("id","date","hour","so2")
conc2 <- reshape(conc1, idvar=c("date","hour"), timevar="id", direction="wide")
colnames(conc2) <- c("date","hour","augusta")
#conc2 <- conc2[,1:4]

```

```

#merge(met2,conc2)
data <- merge(met2,conc2)
colnames(data) <- c("date","hour","ws","wd","augusta")
data1 <- data.frame(substr(data$date,1,4),substr(data$date,5,6),substr(data$date,7,8),data[,2:5])
data1$augusta <- as.numeric(as.character(data1$augusta))*1000
colnames(data1) <- c("yr", "mo", "day", "hr", "ws", "wd","augusta")
data2 <- data.frame(as.POSIXct(paste(data1$yr,"-",data1$mo,"-",data1$day, " ",data1$hr,:00",sep=""),format = "%Y-%m-%d %H:%M:%S"),data1[,5:7])
colnames(data2) <- c("date", "ws", "wd","augusta")
pdf(file="C:\\Users\\ahawki03\\Desktop\\beta_request\\data_from_AQS\\augusta_plots.pdf",width=11,height=8.5)
windRose(data1,angle=5,key.footer="knots",main="1995-1998 Windrose Augusta Quarry 7m")
polarPlot(data1,pollutant='augusta',units="knots",uncertainty = TRUE,resolution="fine")
polarPlot(data1,pollutant='augusta',units="knots",statistic="weighted.mean",resolution="fine",main="SO2 polarPlot (Augusta)")
polarFreq(data1,pollutant='augusta',main="SO2 polarFreq (Augusta)",statistic="weighted.mean",min.bin=2,offset = 5, ws.int = 15, trans = FALSE)
pollutionRose(data1,pollutant='augusta',angle=5,main="SO2 pollutionRose (Augusta)",breaks = c(0,3, 5, 10, 20, 40))
polarCluster(data2, pollutant = "augusta", n.clusters = 2:8,main="SO2 polarCluster (Augusta)")

for (jj in 1995:1998){
for (ii in 1:12){
timePlot(selectByDate(data2, year=jj, month=ii),pollutant = c("augusta", "ws","wd"),main=paste(jj,"Augusta Labadie SO2"))
}}
dev.off()

camd <- read.csv(file="C:\\Users\\ahawki03\\Desktop\\beta_request\\data_from_AQS\\Labadie1995-1998.txt",skip=0,header=T,sep=",")
camd$date <- as.POSIXct(paste(camd$OP_DATE, camd$OP_HOUR),format="%m-%d-%Y %H")
camd1 <- reshape(camd, idvar=c("date"), timevar="UNITID", direction="wide")
camd2 <-
data.frame(camd1$date,camd1$SO2_MASS.1,camd1$SO2_MASS.2,camd1$SO2_MASS.3,camd1$SO2_MASS.4,camd1$HEAT_INPUT.1,camd1$HEAT_INPUT.2,camd1$HEAT_INPUT.3,camd1$HEAT_INPUT.4)
camd2$total <- rowSums(camd2[,2:5],na.rm=TRUE)
colnames(camd2) <- c("date", "U1mass", "U2mass", "U3mass", "U4mass", "U1heat", "U2heat", "U3heat", "U4heat", "total")
data3 <- merge(data2,camd2)
write.table(data3,file="C:\\Users\\ahawki03\\Desktop\\beta_request\\data_from_AQS\\R7_summary_1995-1998.csv",sep=",")

#####
library(gridBase)
library(grid)

pdf(file="C:\\Users\\ahawki03\\Desktop\\beta_request\\data_from_AQS\\barbs.pdf",width=11,height=8.5)
dates <- unique(data$date)
for (j in 1:length(dates)) {
met3 <- data[data$date==dates[j],]
met3$y <- c(1:24)*0-10
met3$x <- c(0:23)
met3 <- na.omit(met3)
if (length(met3$ws)>0) {
with(met3, plot(x, y, ylim=c(-20, 60), xlim=c(0, 24), pch=16,main=dates[j]))}
}

```

```

vps <- baseViewports()
pushViewport(vps$inner, vps$figure, vps$plot)
# Plot
for (i in 1:nrow((met3))) {
  pushViewport(viewport(
    x=unit(met3$x[i], "native"),
    y=unit(met3$y[i], "native"),
    angle=360-(met3$wd[i])))
  wind_barb((met3$ws[i]*10))
  popViewport()
}

popViewport(3)
abline(h = 0)
points(met3$valley,col="red",pch=16)
points(met3$NW,col="blue",pch=16)
}}
dev.off()

#for the historic data
data$augusta <- as.numeric(as.character(data$augusta))*1000 #only do this once!!!

#####
#####this is a subset of days
#####
pdf(file="C:\\Users\\ahawki03\\Desktop\\beta_request\\data_from_AQS\\barbs_augusta.pdf",width=11,height=8.5)
dates <- unique(data$date)
for (j in 1:length(dates)) {
  met3 <- data[data$date==dates[j],]
  met3$y <- as.numeric(met3$hour)*0-10
  met3$x <- as.numeric(met3$hour)-1
  met3 <- na.omit(met3)
  if (length(met3$ws)>0) {
    if (max(met3$augusta)>20) {
      with(met3, plot(x, y, ylim=c(-20, max(met3$augusta)+5), xlim=c(0, 24), pch=16,main=dates[j]))}

  vps <- baseViewports()
  pushViewport(vps$inner, vps$figure, vps$plot)
  # Plot
  for (i in 1:nrow((met3))) {
    pushViewport(viewport(
      x=unit(met3$x[i], "native"),
      y=unit(met3$y[i], "native"),
      angle=360-(met3$wd[i])))
    wind_barb((met3$ws[i]*10))
    popViewport()
  }

  popViewport(3)
  abline(h = 0)
  points(met3$augusta,col="red",pch=16)
  }#end check for met data to draw wind barbs
  }#end max concentration check
}#end for loop
dev.off()

```

```

#####
this is all
days#####
pdf(file="C:\\Users\\ahawki03\\Desktop\\beta_request\\data_from_AQS\\barbs_augusta_all.pdf",width=11,height=8.5)
dates <- unique(data$date)
for (j in 1:length(dates)) {
met3 <- data[data$date==dates[j],]
met3$y <- as.numeric(met3$hour)*0-10
met3$x <- as.numeric(met3$hour)-1
met3 <- na.omit(met3)
if (length(met3$ws)>0) {
with(met3, plot(x, y, ylim=c(-20, 200), xlim=c(0, 24), pch=16,main=paste("Labadie Augusta Quarry", dates[j])))
vps <- baseViewports()
pushViewport(vps$inner, vps$figure, vps$plot)
# Plot
for (i in 1:nrow((met3))) {
  pushViewport(viewport(
    x=unit(met3$x[i], "native"),
    y=unit(met3$y[i], "native"),
    angle=360-(met3$wd[i])))
  wind_barb((met3$ws[i]*10))
  popViewport()
}
popViewport(3)
abline(h = 0)
abline(h = 75,col="red")
points(met3$augusta,col="red",pch=16)
}#end check for met data to draw wind barbs
}#end for loop
dev.off()

#####
# wind barb function from http://stackoverflow.com/questions/32705013/plot-wind-barb-in-r

wind_barb <- function(x, mlength=0.1, wblength=0.025) {

# Calculate which / how many barbs
# any triangles (50)
fif <- floor(x /50)
# and then look for longer lines for remaining speed (10)
tn <- floor( (x - fif* 50)/10)
# and then look for shorter lines for remaining speed (5)
fv <- floor( (x - fif* 50 - tn* 10)/5)

# Spacing & barb length
yadj <- 0.5+mlength
dist <- (yadj-0.5) / 10
xadj <- 0.5+wblength
xfadj <- 0.5+wblength/2

# Create grobs
main_grob <- linesGrob(0.5, c(0.5, yadj ))
```

```

# 50 windspeed
if(fif != 0) {
  fify <- c(yadj, yadj-dist*seq_len(2* fif) )
  fifx <- c(0.5, xadj)[rep(1:2, length=length(fify))]
  fif_grob <- pathGrob(fifax, fify, gp=gpar(fill="black"))
} else {
  fif_grob <- NULL
  fify <- yadj+dist
}

# Ten windspeed
if(tn != 0) {
  tny <- lapply(seq_len(tn) , function(x) min(fify) - dist*c(x, x-1))
  tn_grob <- do.call(gList,
    mapply(function(x,y)
      linesGrob(x=x, y=y, gp=gpar(fill="black")),
      x=list(c(0.5, xadj)), y=tny, SIMPLIFY=FALSE))
} else {
  tn_grob <- NULL
  tny <- fify
}

# Five windspeed
if(fv != 0) {
  fvy <- lapply(seq_len(fv) , function(x) min(unlist(tny)) -dist* c(x, x-0.5))
  fv_grob <- do.call(gList,
    mapply(function(x,y)
      linesGrob(x=x, y=y, gp=gpar(fill="black")),
      x=list(c(0.5, xfadj)), y=fvy, SIMPLIFY=FALSE))
} else {
  fv_grob <- NULL
}

# Draw
#grid.newpage()
grid.draw(gList(main_grob, fif_grob, tn_grob, fv_grob))
}

#####
#trying to write my own function... does not work because of checkPrep not found.
selectByDate1 <- function (mydata, start = "1/1/2008", end = "31/12/2008", year = 2008,
  month = 1, day = "weekday", hour = 1)
{
  vars <- names(mydata)
  mydata <- checkPrep(mydata, vars, "default", remove.calm = FALSE,
    strip.white = FALSE)
  weekday.names <- format(ISOdate(2000, 1, 3:9), "%A")
  if (!missing(start) & !missing(end)) {
    if (length(grep("/", start)) > 0 & length(grep("/", end)) >
      0) {
      start <- as.Date(start, "%d/%m/%Y")
      end <- as.Date(end, "%d/%m/%Y")
    }
    if (length(grep("-", start)) > 0 & length(grep("-", end)) >

```

```

0) {
  start <- as.Date(start)
  end <- as.Date(end)
}
mydata <- subset(mydata, as.Date(date) >= start & as.Date(date) <=
end)
}
if (!missing(year))
  mydata <- mydata[which(year(mydata$date) %in% year),
]
if (!missing(month)) {
  if (is.numeric(month)) {
    if (any(month < 1 | month > 12))
      stop("Month must be between 1 to 12.")
    mydata <- mydata[which(month(mydata$date) %in% month),
]
  }
  else {
    mydata <- subset(mydata, substr(tolower(format(date,
"%B")), 1, 3) %in% substr(tolower(month), 1,
3))
  }
}
if (!missing(hour)) {
  if (any(hour < 0 | hour > 23))
    stop("Hour must be between 0 to 23.")
  mydata <- mydata[which(hour(mydata$date) %in% hour),
]
}
if (!missing(day)) {
  days <- day
  if (is.numeric(day)) {
    if (any(day < 1 | day > 31))
      stop("Day must be between 1 to 31.")
    mydata <- mydata[which(day(mydata$date) %in% days),
]
  }
  else {
    if (day[1] == "weekday")
      days <- weekday.names[1:5]
    if (day[1] == "weekend")
      days <- weekday.names[6:7]
    mydata <- subset(mydata, substr(tolower(format(date,
"%A")), 1, 3) %in% substr(tolower(days), 1, 3))
  }
}
mydata
}

```